2018-09-10

Artibiotics: An Artificial Biology Musical Experiment

Miranda, Eduardo

Biofaction KG

Artibiotics: An Artificial Biology Musical Experiment

Eduardo Reck Miranda
Interdisciplinary Centre for Computer Music Research (ICCMR)
Plymouth University
United Kingdom
eduardo.miranda@plymouth.ac.uk

1 Introduction

Artibiotics is an experimental musical composition for percussion ensemble and electronics. It articulates the development of Artificial Biology, which is a new concept conceived during my residency at Wagner Lab in Regensburg in 2017.

The residency was supported by Biofaction in connection with the European Commission-funded Synpeptide project, aimed at the design of new antibiotics.

Artibiotics will be premiered at the Gala Concert of the Peninsula Arts Contemporary Music Festival, Plymouth, on 03 March 2018, by *Ensemble Bash*.

2 Background

Easy access and misuse of antibiotics are prompting various types of harmful bacteria to developing resistance to existing antimicrobial substances. This is a problem of serious concern. The World Health Organisation has recently announced that the increasing difficulty of curing diseases caused by new drug-resistant bacteria is a threat worldwide. To address this problem, the Synpeptide project is looking into developing new types of lantibiotics, which is a class of antimicrobial molecules that have not been widely explored in antibiotics research yet.

Essentially, Wagner Lab is developing Synthetic Biology work. The team is unravelling the structure and function of naturally occurring lantibiotics with a view on engineering new kinds of lantibiotics. In a nutshell, the research involves shuffling the DNA code of known lantibiotics, synthesise the new molecules and test them *in vivo* against specific kinds of bacteria. This is a laborious and time-demanding process, as the number new possible re-combinations are immense. Wagner Lab developed strategies to optimise this. For instance, given that lanbitiotics' codes follow a modular structure, where each module contributes a distinct characteristic of the molecule, the re-combinations are accomplished at the level of those modules. For example, sections marked as A, B, C, D and E in the structure of shown in Figure 2 are different modules of a lantibiotic known as *Nisin*.

At its most fundamental level a lantobiotic is coded as a strand of the DNA bases: adenine (A), guanine (G), cytosine (C) and thymine (T), where A and G are referred to as purines and C and T are referred to as pyrimidines. The *Nisin* molecule is encoded with 102 DNA bases. However, the DNA representation does not fully represent the actual

molecule per se. It needs to be rendered into a chain of amino acids, which forms a small protein, referred to as a peptide. In fact this process involves 3 stages: transcription, translation and post-translational modifications.

Each triplet of DNA - referred to as a codon - encodes 1 amino acid of the chain. In simplistic terms, firstly DNA is transcribed into RNA, which is subsequently translated into amino acids. For instance, *Nisin* is formed by a chain of 34 amino acids; represented by 34 codons. After translation, chemical reactions take place (e.g., dehydration), resulting in the final peptide.

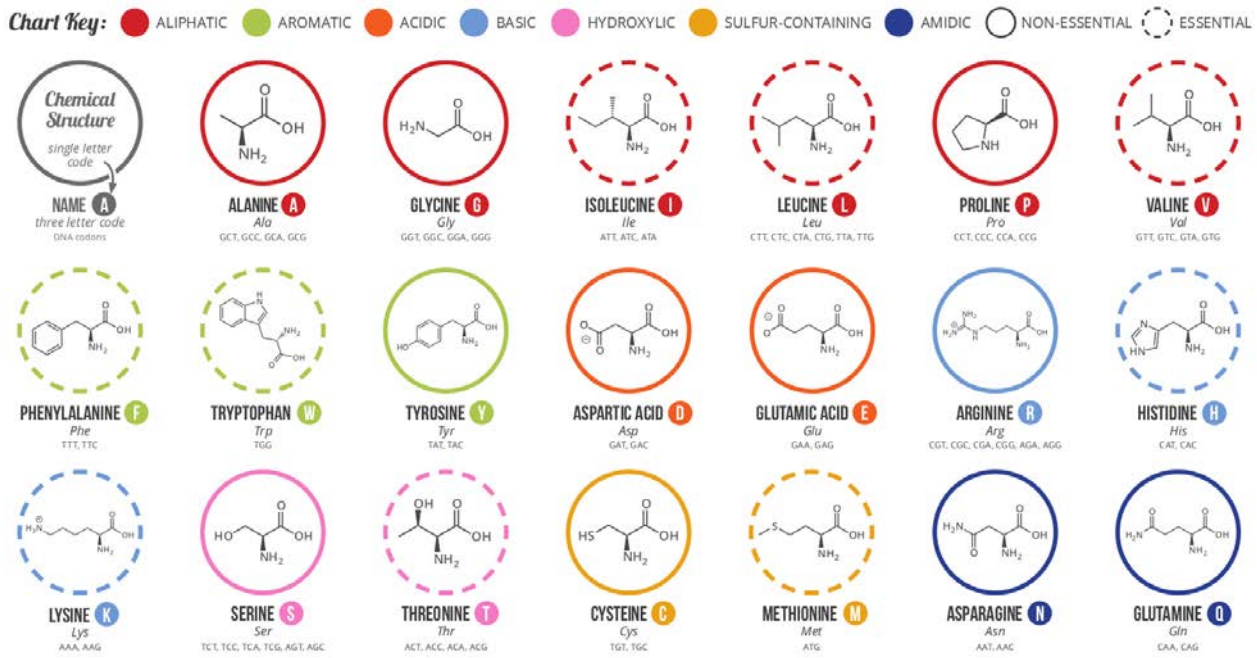


Figure 1: Chart of amino acids. (From the http://www.compoundchem.com/2014/09/16/aminoacids/)

Although there are over 500 different amino acids in nature, only 20 of them is encoded by means of DNA directly; see chart in Figure 1. What is important to understand here is that there are 64 possible DNA codons to encode these 20 basic amino acids, which means that most of them can be encoded by more than 1 codon. For instance, whereas *Glutamine* is encoded by CAA or CAG, *Lysine* is encoded by AAA and AAG (Figure 1).

Figure 2 shows a schematic rendering of *Nisin* from its DNA code. The first animo acid is *Isoleucine* (Ile), which correspond to ATT (of the codon of the DNA strand). The last amino acid is *Lysine* (Lys), corresponding to the last codon: AAA. Note that Dhb and Dha, do not have an associated DNA codon in Figure 1. These two amino acids resulted from post-translational modifications.

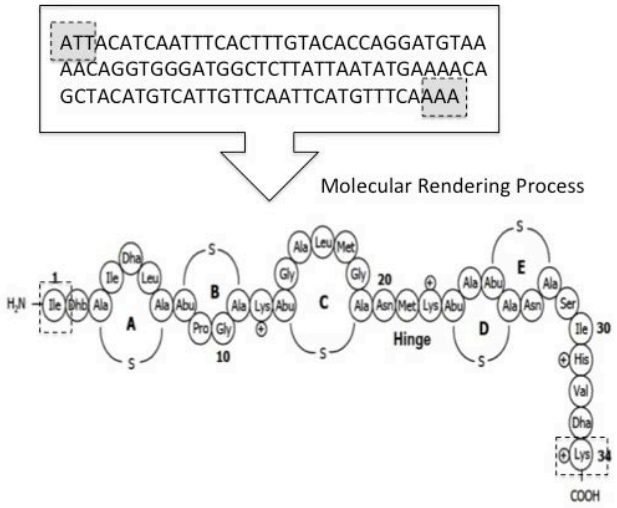


Figure 2: Rendering of *Nisin* from its DNA code.

3 A-Biology

The more I learned about the Synthetic Biology work developed at Wagner Lab during the first days of my residency in Regensburg, the more fascinated I became by the idea of synthesising new chimeric proteins. I wondered if I could take the notion of Synthetic Biology even further, to the point where the very biochemistry of Biology could become chimeric, blending musical and biological processes.

In many ways, the scientific modus operandi that I witnessed in the laboratories in Regensburg reminded me of my own creative modus operandi. I felt that the process of synthesising new proteins is not so different from the process of creating new pieces of music. This led to the invention of a new concept: *Artificial Biology*, or *A-Biology*, which is the surreal biology of a parallel universe of musical molecules.

The A-Biology framework consists of a number of pieces of software[1] that processes strands of DNA, including:

- Miranda machine
- Rhythmator
- Pitch-folding
- Post-translational musifications

3.1 Miranda machine

At the core of A-Biology is the Miranda machine, which is an abstract Turing machine-like processor[2]. Inspired by the Turing machine, which is an abstract machine that manipulates a sequence of symbols according to a set of rules, the Miranda machine

[1] I am thankful to ICCMR student Satvik Venkatesh who kindly helped to develop the software.
[2] The Turing machine is a mathematical model of a hypothetical computer, invented by the mathematician Alan Turing in 1936.

was designed to manipulate sequences of DNA strands: it transcribes a DNA strand into a sequence of commands, or a DNA program[3].

Given a DNA strand, the Miranda machine produces a chain of data processing commands. In standard Biology this process would have produced a chain of amino acids, which would form an enzyme. In A-Biology, however, this chain of commands forms a program that processes the originating DNA strand itself. In other words, the Miranda machine transcribes a DNA code into a program to modify its own code.

In standard Biology, the resulting enzymes would operate on DNA strands. Similarly, in A-Biology each of the 20 basic amino acids is associated with a command of Miranda machine's programming language, as shown in Table 1. Each command has an orientation, S, L or R, the meaning of which will be explained later.

The machine parses a DNA strand in groups of three consecutively occurring bases, or codons. For instance, consider the following DNA strand consisting of 15 codons:

ATGAACGCGGAGAGGATTTGTCGCTGGCCTTAGTATCATTCCAAA

The derived DNA program for the strand above is as follows:

cop - **rpy** - **ina** - **lpu** - **swi** - **ing** - **cut** - **swi** - **rev** - **adl** - **off** - **ivl** - **deb** - **mvr** - **dec**

For example ATG is transcribed as **cop**, AAC as **rpy**, GCG as **ina**, and so forth. If a strand finishes with an incomplete codon (i.e., with 1 or 2 bases) than the parser will ignore them.

The application of the DNA program to process the originating DNA strand results in the following set of five new DNA strands:

1. CATATTGTCGCTGGCCTTAAGTATCATTCCAAA
2. TATG
3. GAGAGGCGGTA
4. C
5. CCT

See Appendix for a step-by-step run of the example above. Effectively, the Miranda machine is normally set to chain reaction mode. In this case it will continue processing the newly generated DNA strands, which will subsequently generate new sets of strands and will carry on processing the new sets, and so on, until a given halting criterion is met. The end result will be a pool of DNA strands, consisting of the original strand and the groups of strands derived from each cycle.

[3] The Miranda machine builds upon the notion of Typogenetics, introduced by Douglas R. Hosftadter in his book *Gödel, Escher, Bach: An Eternal Golden Braid* (Penguin Books, 1979).

Amino acid	Command	Codons	Action
Serine	mvr[S]	TCT, TCC, TCA, TCG, AGT, AGC	Move right by one unit
Threonine	mvl[S]	ACT, ACC, ACA, ACG	Move left by one unit
Alanine	ina[S]	GCT, GCC, GCA, GCG	Insert base A to the right
Glycine	inc[L]	GGT, GGC, GGA, GGG	Insert base C to the right
Isoleucine	ing[L]	ATT, ATC, ATA	Insert base G to the right
Leucine	int[R]	CTT, CTC, CTA, CTG, TTA, TTG	Insert base T to the right
Proline	adl[R]	CCT, CCC, CCA, CCG	Insert random base to the left
Valine	adr[L]	GTT, GTC, GTA, GTG	Insert random base to the right
Methionine	cop[L]	ATG	Enable copy mode
Cysteine	cut[S]	TGT, TGC	Cut strand
Arginine	swi[L]	CGT, CGC, CGA, CGG, AGA, AGG	Switch enzyme to another strand
Histidine	deb[S]	CAT, CAC	Delete base
Lysine	dec[S]	AAA, AAG	Delete codon
Asparagine	rpy[L]	AAT, AAC	Search for pyrimidine to the right
Aspartic acid	rpu[R]	GAT, GAC	Search for purine to the right
Glutamine	lpy[R]	CAA, CAG	Search for pyrimidine to the left
Glutamic acid	lpu[R]	GAA, GAG	Search for purine to the left
Phenilalanine	ivr[R]	TTT, TTC	Invert base to the right
Tyrosine	ivl[L]	TAT, TAC	Invert base to the left
Tryptophan	rev[S]	TGG	Perform lateral inversion on strand
	off[R]	TAA, TAG, TGA	Disable copy mode

Table 1: The Miranda machine's commands associated to amino acids, their respective codons and action.

3.2 Rhythmator

The Rhythmator translates a given DNA strand into a rhythmic sequence. A-Biology includes a scheme to represent amino acids by means of distinct vocabularies of four rhythmic figures, referred to as *nucleo-rhythms*, each of each associated with the bases A, T, G and C. This enables to the Rhythmator to parse a DNA strand and translate its codons into *rhythmic codons*.

For instance, Figure 3 shows a vocabulary of nucleo-rhythms and Figure 4 shows the respective rhythmic codons for the amino acids Cysteine (TGT and TGC) and Methionine (ATG).

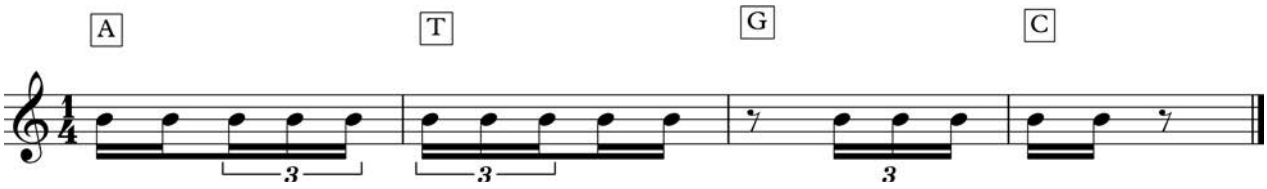


Figure 3: An example of a vocabulary of nucleo-rhythms.

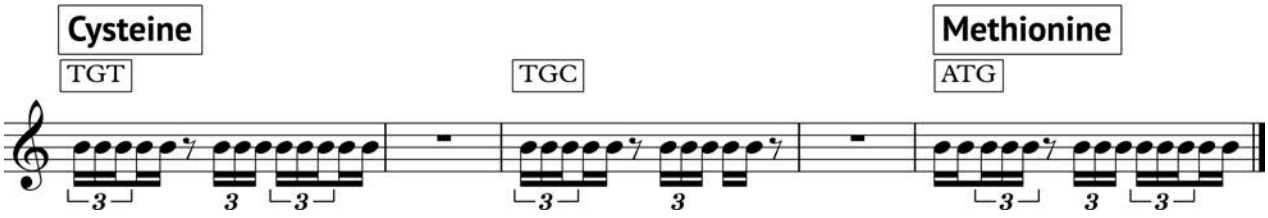


Figure 4: Rhythmic codons for Cysteine and Methionine.

In standard Biology, amino acids are classified according to their chemical structure: Aliphatic, Aromatic, Acidic, Basic, and so on. In A-Biology, each of these classes is represented by a different vocabulary of nucleo-rhythms. For instance, the vocabulary in Figure 3 is used for Sulfur-containing amino acids.

Nucleo-rhythms vocabularies are built either manually or automatically. The one showed in Figure 3 was built manually; that is, I composed these nucleo-rhythms myself. However, I also generated vocabularies automatically for *Artibiotics*. In order to do this I devised a piece of software that extracts basic rhythmic figures from given musical scores.

I generated a number of vocabularies from Brazilian samba for *Artibiotics*. An example of a samba vocabulary is shown in Figure 5 and samba rhythmic codons for 3 Aliphatic amino acids are shown in Figure 6.

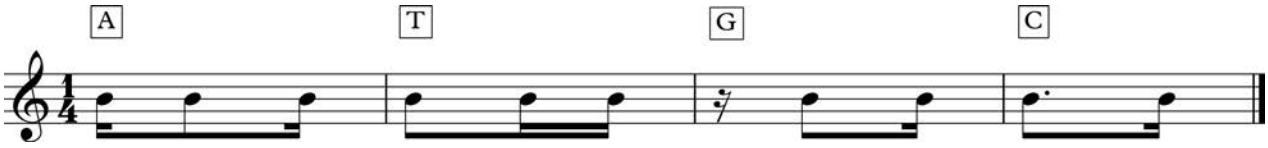


Figure 5: An example of a vocabulary of nucleo-rhythms extracted from samba music.

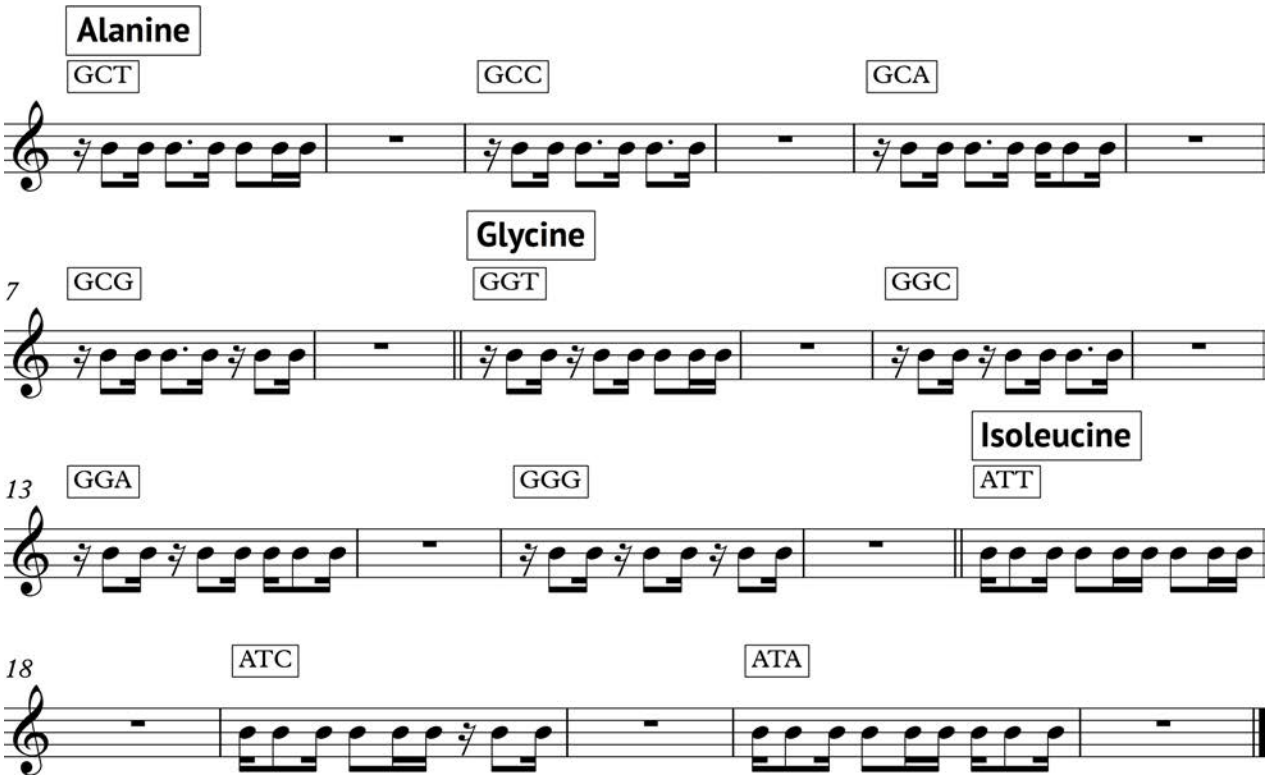


Figure 6: Samba rhythmic codons for 3 Aliphatic amino acids.

3.3 Pitch-folding

Pitch-folding is inspired by the phenomenon of protein folding. Whereas in standard Biology protein folding defines the shape of a molecule, in A-Biology pitch-folding defines pitches for a rhythmic sequence. The pitch-folding phase is optional, depending on weather one wishes to assign pitches to the sequences of not. Indeed, pitch-folding was not always used for the composition of *Artibiotics* because the piece is for pitched and non-pitched percussion instruments.

If the pitch-folding mechanism is activated, then the Miranda machine takes into account the orientation information of DNA commands (represented inside brackets in Table 1) to "fold" the derived program. In order to visualise this process, consider each DNA command in Table 1 as a block, with the command name and its orientation in the

centre, and an arrow pointing in the direction corresponding to its orientation. A command can have one three possible orientations: right, straight and left, represented as R, S and L, respectively; e.g., **rpu[R]**, **mvr[S]** and **swi[L]** (Figure 7).

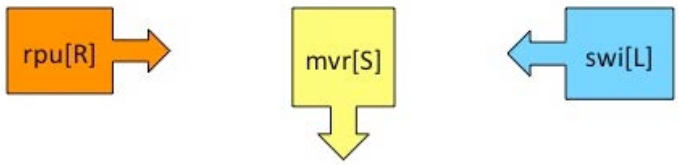


Figure 7: Visualisation of a command as block and its orientations.

The system produces a domino-like chain of blocks, such that each block is connected to the top of the next block (Figure 8).

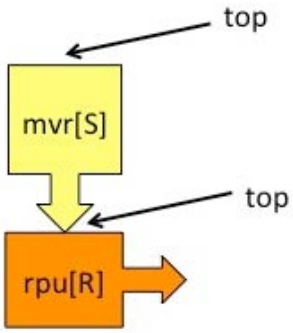


Figure 8: Connection between 2 blocks

As an example, let us consider the hypothetical strand TCTGGTGTACCCCAG. This strand gives us the following sequence of commands and orientations: **mvr[S] - inc[L] - adr[L] - adI[R] - lpy[R]**. The resulting folded block sequence is depicted in Figure 9.

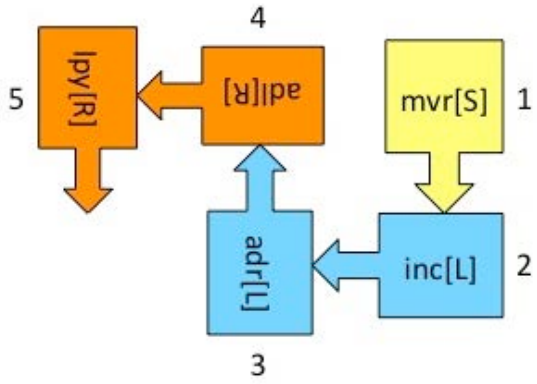


Figure 9: Visual representation of a folded block sequence of DNA programming commands.

Once the command sequence is folded, the system establishes the cardinal direction of the command: north (NO), south (SO), east (EA) and west (WE), respectively. In the case of the above example, the sequence **mvr[S] - inc[L] - adr[L] - adI[R] - lpy[R]** becomes **mvr[SO] - inc[WE] - adr[NO] - adI[WE] - lpy[SO]**.

Every DNA programming command is associated with 4 distinct algorithms for generating pitches, one for each cardinal direction. An example is given in Table 2.

Command	**Generative Pitch Algorithm**
swi[WE]	*s* = *r* – 7 ;transpose one major 5th down *x* = 1 for each note, do while there are notes to process { make_note(*s*) *u* = *s* + *x* make_note(*u*) *d* = *s* – *x* make_note(*d*) *x* = *x* + 1 }
swi[EA]	*s* = *r* – 7 ; transpose one major 5th down *x* = 1 for each note, do while there are notes to process { make_note(*s*) *d* = *s* – *x* make_note(*d*) *u* = *s* + *x* make_note(*u*) *x* = *x* + *1* }
swi[NO]	*s* = *r* – 7 ; transpose one major 5th down *x* = *1* for each note, do while there are notes to process { make_note(*s*) *u* = *s* + *x* make_note(*u*) *x* = *x* + 1 }
swi[SO]	*s* = *r* – 7 ; transpose one major 5th down *x* = *1* for each note, do while there are notes to process { make_note(*s*) *d* = *s* – *x* make_note(*d*) *x* = *x* + 1 }

Table 2: Generative pitch algorithms for command swi. The variable *r* is for a pre-established reference pitch; that is, a MIDI note number (e.g., 60 corresponds to Middle C).

As an example of the A-Biology generative process let us consider the DNA strand for a lantibiotic referred to as *Columbicin A*:

GCTGGACGTGGATGGATTAAAACACTTACAAAAGATTGTCCAAATGTTATTTCATCA
ATTTGTGGAACAATTATTACAGCTTGTAAAAATTGTGCT

The first cycle of the Miranda machine applied to *Columbicin A* produced the following set of strands:

1. TCGTGTTAAAAATGTTCGACATTATCGCTACAAGGTGTTTAACTACTTTAT
   TGTAAACCTGTTAGAAAACATCGATGGGAA

2. AAATTAGGTAGGTGCAGGTCG

3. GC

Let us focus on strand number 2. The outcome from applying the rhythmator to strand number 2 is shown in Figure 10.

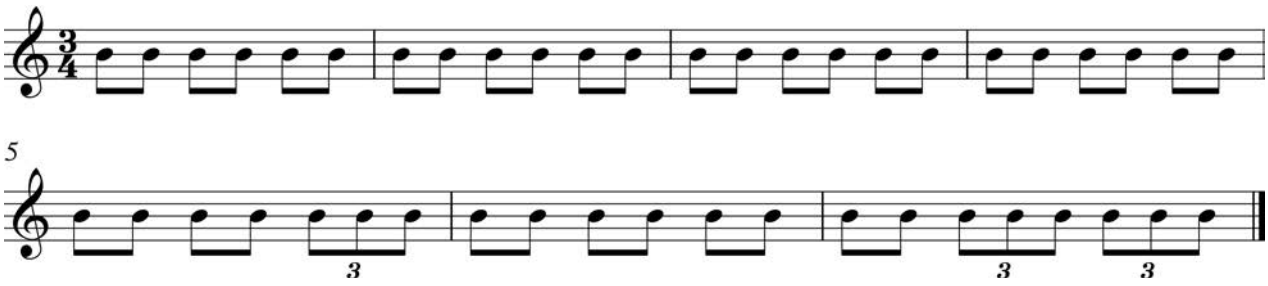


Figure 10: The rhythmic sequence yielded by DNA strand number 2.

If we apply pitch-folding to strand number two, then the resulting programming sequence with cardinal orientation information would be as follows:

dec(SO) - int(EA) - inc(SO) - swi(WE) - cut(WE) - swi(NO) - mvr(NO)

Figure 11 shows the rhythmic sequence with the respective pitch information; in this case the reference pitch is Middle C, or MIDI note 60.

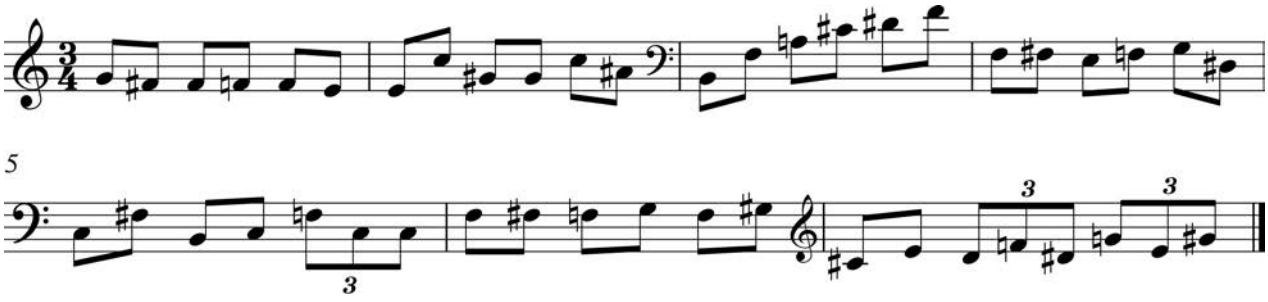


Figure 11: The rhythmic sequence yielded by DNA strand number two with pitch-folding.

4 Post-translational musifications

Post-transaltional musifications are modifications made on the resulting musical sequences generated by the processes described above. The results from these modifications are referred to as *musical molecules*.

Post-translational musifications involve processes that might be applied to the sequences automatically (e.g., the A-Biology equivalent of dehydration in standard Biology) and manually. For the composition or *Artibiotics* post-translational musifications were kept to a minimum and were applied mostly to those sequences that were scored for the percussionists to perform; for example, I transposed pitches that would not have been possible to play by a certain instrument. Those sequences played electronically were rarely modified; that is, they did not need post-translational musifications to become musical molecules.

Post-translational musifications gave me opportunity to exercise my aesthetic judgement and adapt the sequences to produce musical molecules for specific musical contexts as I put these materials together to compose the piece.

5 The composition process

The composition process involved two stages: generation of musical molecules with the A-Biology system introduced above and assemblage of the piece. The latter occasionally required further editing of the musical molecules to fit practical requirements such as playing techniques, dynamics, and so on.

Artibiotics has three movements: *Pathostuff*, *Trials* and *Musicin*. The piece begins by exposing a pathogen that needs to be eliminated. Then, the second movement portrays a battle to kill the pathogen. Various lantibiotics are tried until one of them prevails. The third movement celebrates the successful lantibiotic.

For the first movement I run the A-Biology system on the DNA strand of a hypothetical pathogen to generate a relatively large set of musical molecules, referred to as *pathostuff*, hence the name of the movement.

For the second movement I ran the system on the DNA of lantibiotics routinely used for research at Wagner Lab (e.g., *Nisin*, *Pep5*, *Lactosin S*, *Columbicin A*, etc.) and on the DNA strands of *articins.* Articins are chimeric lantibiotics of my own design using DNA information from Wagner Lab's library of lantibiotics. For instance the following is *Articin 5*, consisting of 114 DNA bases, which I designed by combining DNA modules from *Lactosin S* and *Nisin*:

TCAACACCAGTTTTAGCTTCAGTTGCTGTTTCAATGGAACTTCTTCCAACAGCTTCA
GTTCTTTATACAGGTGCTCTTATGGGATGTTTTAAATATTCAGCTAAACATCATTGT

Musically, while pathotuff materials are developed throughout the second movement, 20 contrasting lantibiotic musical molecules emerge, one after another, as if they were attempting to eradicate pathostuff musical molecules. Towards the end of the movement, one of the lantibiotics prevails and pathostuff music finally fades out.

The third movement presents a scherzo with musical molecules associated with the winning lantibiotic.

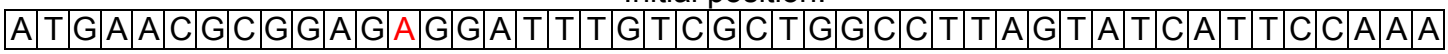
Appendix: A detailed step-by-step run of the Miranda machine

Input strand: ATGAACGCGGAGAGGATTTGTCGCTGGCCTTAGTATCATTCCAAA

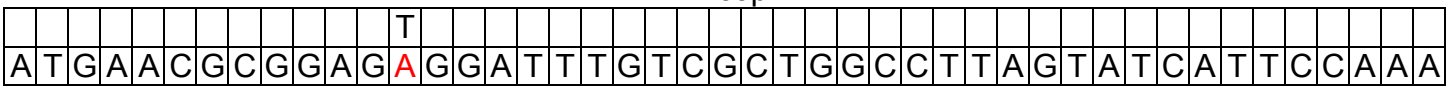
Derived DNA programming code: **cop – rpy – ina - lpu - swi - ing - cut - swi - rev - adl - off - ivl - deb - mvr - dec**

By default, the Miranda machine starts on a randomly selected adenine.

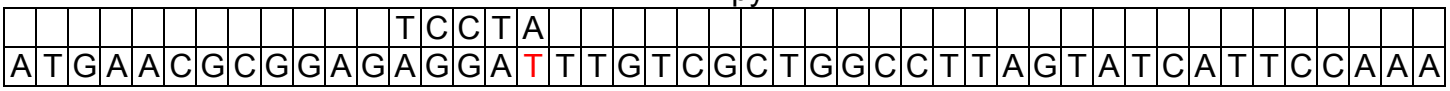
Initial position:



1. cop

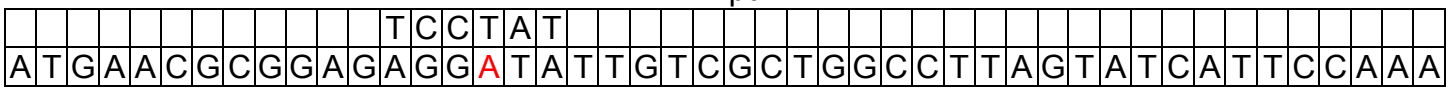


2. rpy

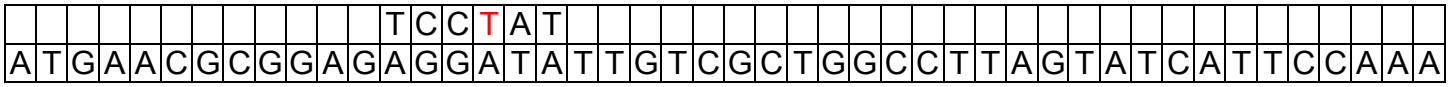


3. ina

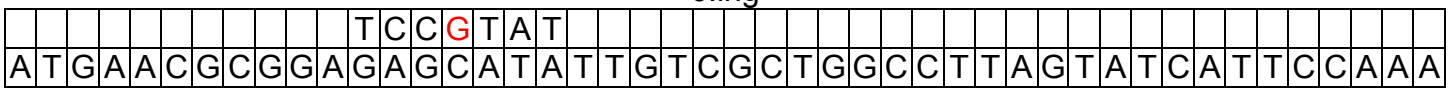
4. lpu



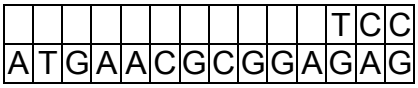
5. swi

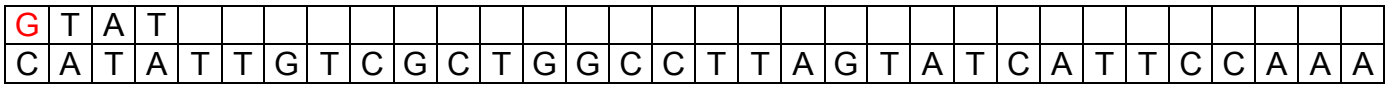


6.ing

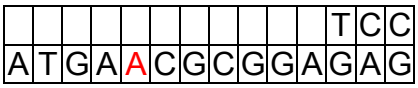


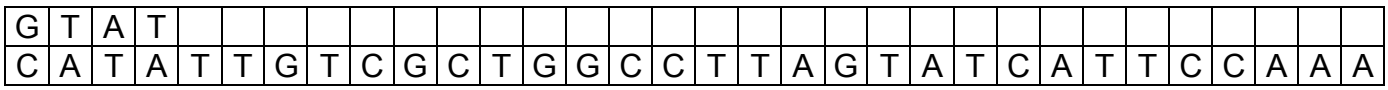
7. cut





8. swi





9. rev

T	C	C											
G	A	G	A	G	G	C	G	C	A	A	G	T	A

G	T	A	T																									
C	A	T	A	T	T	G	T	C	G	C	T	G	G	C	C	T	T	A	G	T	A	T	C	A	T	T	C	C	A	A	A

10. adl

T	C	C						C					
G	A	G	A	G	G	C	G	C	G	A	A	G	T	A

G	T	A	T																									
C	A	T	A	T	T	G	T	C	G	C	T	G	G	C	C	T	T	A	G	T	A	T	C	A	T	T	C	C	A	A	A

11. off

T	C	C						C					
G	A	G	A	G	G	C	G	C	G	A	A	G	T	A

G	T	A	T																									
C	A	T	A	T	T	G	T	C	G	C	T	G	G	C	C	T	T	A	G	T	A	T	C	A	T	T	C	C	A	A	A

12. ivl

T	C	C						C					
G	A	G	A	G	G	C	G	G	G	A	A	G	T	A

G	T	A	T																									
C	A	T	A	T	T	G	T	C	G	C	T	G	G	C	C	T	T	A	G	T	A	T	C	A	T	T	C	C	A	A	A

13. deb

T	C	C						C					
G	A	G	A	G	G	C	G	G	A	A	G	T	A

G	T	A	T																									
C	A	T	A	T	T	G	T	C	G	C	T	G	G	C	C	T	T	A	G	T	A	T	C	A	T	T	C	C	A	A	A

13. mvr

T	C	C						C					
G	A	G	A	G	G	C	G	G	A	A	G	T	A

G	T	A	T																									
C	A	T	A	T	T	G	T	C	G	C	T	G	G	C	C	T	T	A	G	T	A	T	C	A	T	T	C	C	A	A	A

14. dec

T	C	C						C		
G	A	G	A	G	G	C	G	G	T	A

G	T	A	T																									

C	A	T	A	T	T	G	T	C	G	C	T	G	G	C	C	T	T	A	G	T	A	T	C	A	T	T	C	C	A	A	A

The resulting DNA strands are:

1. CATATTGTCGCTGGCCTTAAGTATCATTCCAAA
2. TATG
3. GAGAGGCGGTA
4. C
5. CCT